目錄

文件系统层次结构

本书是《Filesystem Hierarchy Standard》一书的中文译本。该文档规定了**Unix/Linux**系统中各层次目录的作用，是发行版规划目录层次的指南。对于**Mac OS**和**Android**等系统，也有很大指导意义。

当前对应FHS版本为2.3, 于2004年发布。

原始稿完成于作者大学时期。由**WPS**写作完成。在此使用**Gitbook**转为在线版本。方便管理更新。希望能帮助更多的人走进 **UNIX/Linux** 系统世界。

本人水平有限，翻译中难免有不足之处。欢迎向我提出问题和建议。

版本历史：

> - 2014.06.21 初始版本
> - 2014.09.08 将尾注改写为markdown原生格式。修正部分错误。
> - 2014.10.30 迁移到github

作者：

> Wylmer Wang <wantinghard+@+gmail_com>

在此阅读这本书.

> 如果您觉得这份工作对自己有帮助，欢迎表达支持，包括赞助。
>
> 

1. 介绍

目的

本标准使得：

- 软件可以预测已安装文件和文件夹的位置；
- 用户可以预测已安装文件和文件夹的位置。

我们的做法是：

- 指定文件系统各个区域的指导性原则；
- 指定最少应该包含的必需的文件和文件夹；
- 列举原则中的例外情况；以及
- 列举有历史冲突的特殊情况。

FHS 文档使用者包括：

- 创建与FHS兼容并在与FHS兼容发行版中运行应用程序的独立软件提供商；
- 提供与FHS兼容系统的操作系统创建者；以及
- 理解和维护系统的FHS兼容性的用户。

FHS文档适用于有限的范围：

- 本地文件的本地放置位置是一个因地制宜的事情，FHS不会试图强制系统管理员。
- 当文件放置位置需要多个组织，如本地网站、发行版、应用程序、文档等等，之间协调时，FHS会列出问题。

约定

我们建议您阅读本文档的格式化版本，而非纯文字版本。在格式化版本中，文件和文件夹的名称以等宽字体[1]显示。

文件名中可变的部分用“`<`”“`>`”括起来的部分表示，如`<thus>`。电子邮件地址也用“`<`”“`>`”括起来，不过使用正常字体。

文件和文件夹名的变量子字符串用“`*`”表示。

文本中标记为

> 基础知识

的部分是解释说明，不是标准的内容。

---

[1] 网页版中使用代码标记`code`显示。

2. 文件系统

本标准假定使用与FHS兼容文件系统的操作系统也支持在绝大多数UNIX文件系统中使用的基本安全特性。

对于文件之间的区别，可以有两种定义：可共享与不可共享文件、可变与静态文件。通常，在这些方面有区别的文件应该放置在不同的文件夹中。这样在不同的文件系统上保存不同用途的文件会很方便。

“可共享”文件是可以保存在一个主机上而在另外的主机上使用的文件。“不可共享”文件是那些不能共享的文件。例如，用户主文件夹下的文件是可共享的，而设备锁文件是不可共享的。

“静态”文件包括二进制文件、库文件、文档和其他在系统管理员不介入就不会变动的文件。“可变”文件是非静态的文件。

> ## 基础知识
>
> 可共享文件可以保存在一个主机上并在其他多个主机上使用。而典型情况下，文件系统层次结构中不是所有的文件都是可以共享的，因此每个系统具有至少保存了自己不共享文件的本地存储空间。如果一个系统所需要的文件都保存在另一个主机上，通过挂载该主机的一个或多个目录来访问它们会很方便。
>
> 静态和可变文件应该被分开保存，因为静态文件与可变文件不同，它们可以保存上只读的媒质上，并且不需要像调度可变文件那样需要备份。
>
> 历史上UNIX风格的文件系统层次结构在 `/usr` 和 `/etc` 文件夹下既保存静态文件，也保存可变文件。为了实现上面提到的优点，发明了 `/var` 层次结构，所有 `/usr` 下面的可变文件被转移到 `/var` 下面。因而 `/usr` 现在可以只读的方式挂载(如果它在单独的文件系统上)。 `/etc` 下的可变文件早已转移到 `/var` ，因为技术上允许。
>
> 这里是一个与FHS兼容系统的例子。(也可能存在其他与FHS兼容的布局)
>
> | | 可共享 | 不可共享 |
> |---|---|---|
> | 静态 | /usr | /etc |
> | | /opt | /boot |
> | 可变 | /var/mail | /var/run |
> | | /var/spool/news | /var/lock |

根文件系统

用途

根文件系统的内容必须足以启动、还原、恢复和/或修复系统。

要启动系统，根分区上必须存在足以挂载其他分区的内容。这包括工具、配置、引导装载程序信息和其他启动必需的数据。 `/usr` 、 `/opt` 和 `/var` 的设计使得它们可以位于其他的分区或文件系统。

要能够恢复和/或修复系统，一位有经验的维护者诊断和重建损坏系统所需要的工具必须放置在根文件系统中。

要还原系统，从系统备份(软盘、磁带等等)还原的工具必须放置在根文件系统中。

> ## 基础知识
>
> 平衡这些考虑——即倾向于在根文件系统上放置许多文件——的要点是，让根系统系统尽量小。有许多原因要求保持根文件系统尽量小：
>
> 有时需要从很小的媒质上挂载它；
>
> 根文件系统上包含了许多系统特有的配置文件。可能有系统特定的内核、特定的主机名等等。这意味着根文件系统不总是可以在网络化的系统间共享的。在网络化系统的服务器上保持小的根文件系统能够使不可共享文件导致的空间损失降到最小。它也允许工作站拥有更小的本地硬盘。
>
> 尽管您或许可以把根文件系统放置在一个大的分区上，并且也足以往内里随意填充内容，但还有一些用户，他们的分区很小。如果您安装了更多文件，那么可能会发现与其他在较小分区上使用根文件系统的系统不兼容。如果您是一位开发者，那您的假定会成为众多用户的一个问题。
>
> 根文件系统上的磁盘错误所导致的数据损坏导致的问题相比其他分区更严重。小的根文件系统更不容易因系统崩溃而损坏。

应用程序任何时候都不能创建或要求在根文件夹下保存特殊文件或子文件夹。FHS层次结构中规定的其他位置为任何软件包（的这些要求）提供了足够的灵活性。

> ## 基础知识
>
> 禁止在根文件系统上创建新的子文件夹有许多原因：
>
> - 它要求根分区上的空间，而系统管理员可能想保持根分区较小和简单，为了性能或安全的原因。
> - 它逃避了系统管理员设立的如何在可挂载卷上分配标准文件的所有可能规则。
> - 未经对改动后果包括应用程序可移植性的慎重考虑，发行版不应该在根层次结构下创建新目录。

要求

`/` 文件夹下要求有如下文件夹或符号链接。

文件夹	描述
bin	必需命令的二进制文件
boot	引导加载程序的静态文件
dev	设备文件
etc	主机特定的系统配置
lib	必需的共享库和内核模块
media	可移动媒质的挂载点
mnt	临时挂载一个文件系统用的挂载点
opt	外部应用程序软件包
sbin	必需的系统级二进制文件
srv	本系统所提供服务的数据文件
tmp	临时文件
usr	二级层次结构
var	变量数据

以上列出的每个文件夹的具体规范在下面各小节细述。`/usr` 和 `/var` 各由完整的一节来叙述，因为它们比较复杂。

特殊选项

如果相应的子系统已经安装，则 `/` 下必须有以下文件夹或符号链接：

文件夹	描述
home	用户主文件夹（可选）
lib<qual>	其他格式的必需共享库(可选)
root	root用户的主文件夹(可选)

上面列出的每个文件夹的具体规范在下面的小节中细述。

/bin：必需的用户命令二进制文件（所有用户使用）

3.4.1 用途

`/bin` 包含了系统管理员和用户都可能使用的命令，并且是那些要求在没有挂载其他文件系统的情况下能够使用（例如在单用户模式下）的命令。它也可能包含脚本间接使用的命令[1]。

3.4.2 要求

`/bin` 下面不能有子文件夹。

`/bin` 下要求有如下命令。

命令	描述
cat	将文件连接到标准输出的工具
chgrp	改变文件所在组的工具
chmod	改变文件访问权限的工具
chown	改变文件所有者和组的工具
cp	复制文件和文件夹的工具
date	打印或设置系统日期和时间的工具
dd	转换和复制文件的工具
df	报告文件系统磁盘空间使用情况的工具
dmesg	打印或控制内核消息缓存区的工具
echo	显示一行文本的工具
false	什么也不做，只返回不成功的工具
hostname	显示或设置系统主机名的工具
kill	向进程发送消息的工具
ln	在文件之间创建链接的工具
login	在系统上开启会话的工具
ls	列出文件夹内容的工具
mkdir	创建文件夹的工具
mknod	创建块或字符特殊文件的工具
more	按页翻看文本的工具

mount	挂载文件系统的工具
mv	移动/重命名文件的工具
ps	报告进程状态的工具
pwd	打印当前工作文件夹名称的工具
rm	删除文件或文件夹的工具
rmdir	删除空文件夹的工具
sed	‘sed’流编辑器
sh	Bourne命令Shell
stty	改变和打印终端行设置的工具
su	改变用户ID的工具
sync	清洗文件系统缓存区的工具
true	什么也不做，只返回成功的工具
umount	卸载文件系统的工具
uname	打印系统信息的工具

如果 `/bin/sh` 不眞是Bourne shell，它必须为指向眞正shell命令的硬链接或软链接。

`[` 和 `test` 命令必须一起放置在/bin或/usr/bin中。

> 基础知识
>
> 例如bash以 `sh` 或 `bash` 名称调用时表现会有差异。使用符号链接也让用户容易看出 `/bin/sh` 不眞是Bourne shell。
>
> 对 `[` 和 `test` 必须以二进制形式(即使它们是由shell内部实现的)包含进来的要求与POSIX.2相一致。

3.4.3 特殊选项

如果安装了对应的子系统，/bin下必须有以下程序或指向它们的符号链接：

命令	描述
csh	C shell（可选）
ed	‘ed’编辑器（可选）
tar	tar档案工具（可选）
Cpio	cpio档案工具（可选）
gzip	GNU压缩工具（可选）
gunzip	GNU解压工具（可选）
zcat	GNU解压工具（可选）
netstat	网络统计工具（可选）
ping	ICMP网络测试工具（可选）

如果 `gunzip` 和 `zcat` 程序存在，则它们必须是指向 `gzip` 的符号链接或硬链接。 `/bin/csh` 可以是指向 `/bin/tcsh` 或 `/usr/bin/tcsh` 的符号链接。

> ## 基础知识
>
> 为了能够备份系统，添加了 `tar` 、 `gzip` 和 `cpio` 命令（在 `/` 完整的前提下）
>
> 反之，如果不会还原根分区，则可以忽略这些二进制文件（例如， `/` 在ROM芯片上，通过NFS挂载 `/usr` 的情况）。如果计划通过网络还原系统，则 `ftp` 或 `tftp` （及所有其他ftp连接所需的东西）应该放置在根分区中。

> 1. 不是太关键的命令二进制文件必须转而放置在/usr/bin中。 只有非root用户才需要的项目（如X窗口系统、chsh等等）通常不太必要放置在根分区。 ↩

/boot：引导装载程序的静态文件

3.5.1. 用途

此文件夹包含了启动过程中所需的所有文件，除了启动时不需要的配置文件和映射安装程序。因此/boot中存储了内核在开始执行用户模式程序之前所使用的数据。这会包括保存的主启动扇区和扇区映射文件[2]。

> 引导装载程序的配置文件必须放置在/etc中。
>
> GRUB引导装载程序在启动前读取它的配置文件，因而它应该放置在/boot下。但是，它又是一个配置文件，所以应该旋转在/etc下。解法办法是符号链接，如/etc/grub/menu.lst->/boot/menu.lst。

3.5.2. 特殊选项

操作系统内核必须位于 `/` 或者 `/boot` [3]。

> 2. 为引导装载程序启动一个文件作准备所必需的程序必须放置在/sbin中。 ↩

> 3. 在一些i386机器上，由于硬件限制， `/boot` 可能需要放在完全位于启动设备1024柱面以下的独立分区上。</br>一些MIPS系统要求 `/boot` 分区是一个挂载的MS-DOS文件系统或其他固件能访问的文件系统。这可能导致 `/boot` 中可使用的的文件名上有一些限制（只针对受影响的系统）。 ↩

/dev：设备文件

3.6.1. 用途

`/dev` 文件夹是放置特殊或设备文件的地方。

3.6.2. 特殊选项

如果有 `/dev` 下的文件需要手动创建的情况，`/dev` 必须包含一条叫做 `MAKEDEV` 的命令，它可能创建所需的设备。它也可以包含一个用于任何本地设备的叫 `MAKEDEV.local` 的节点。

如果需要，`MAKEDEV` 必须提供创建系统中能发现的所有设备的功能，而不仅仅是那些一种特定实现所安装的设备文件。

/etc：主机特定的系统配置

3.7.1. 用途

`/etc` 层次结构包含了配置文件。“配置文件”是一个用于控制程序操作的本地文件；它必须是静态文件而且不能是可执行的二进制文件[4]。

3.7.2. 要求

`/etc` 下面不能放置二进制文件[5]。

`/etc` 下应该有以下文件夹或指向相应文件夹的符号链接：

文件夹	描述
opt	`/opt` 的配置
X11	X窗口系统的配置（可选）
sgml	SGML的配置（可选）
xml	XML的配置（可选）

3.7.3.特殊选项

如果安装了相应子系统，`/etc` 下必须有以下文件夹或指向文件夹的符号链接：

文件夹	描述
opt	`/opt` 的配置

如果如果安装了相应的子系统，/etc下必须有以下文件或符号链接[6]：

文件	描述
csh.login	系统范围的C shell登录初始化文件（可选）
exports	NFS文件系统访问控制表（可选）
fstab	文件系统的静态信息（可选）
ftpusers	FTP后台服务用户访问控制表（可选）
gateways	路由用的网关列表文件（可选）
gettydefs	`getty` 用的速率和终端设置（可选）
group	用户组文件（可选）
host.conf	解析器配置文件（可选）

hosts	有关主机名的静态信息（可选）
hosts.allow	TCP wrapper的主机访问文件（可选）
hosts.deny	TCP wrapper的主机访问文件（可选）
hosts.equiv	rlogin 、 rsh 、 rcp 的信任主机表（可选）
hosts.lpd	lpd 的信任主机表（可选）
inetd.conf	inetd 的配置文件（可选）
inittab	init 的配置文件（可选）
issue	预登录消息和认证文件（可选）
ld.so.conf	搜索共享库的附加文件夹列表（可选）
motd	当天的登录后消息文件（可选）
mtab	文件系统的动态信息（可选）
mtools.conf	mtools 的配置文件（可选）
networks	网络名称的静态信息（可选）
passwd	密码文件（可选）
printcap	lpd 打印机功能数据库（可选）
profile	系统范围的sh shell登录初始化文件（可选）
protocols	IP 协议列表（可选）
Resolv.conf	解析器配置文件（可选）
rpc	RPC协议列表（可选）
securetty	root登录的TTY访问控制（可选）
services	网络服务的端口名称（可选）
shells	有效的登录shell的路径名称（可选）
Syslog.conf	syslogd 的配置文件（可选）

mtab 不符合 /etc 的静态特性：保留它是因为历史原因[7]。

3.7.4./etc/opt：/opt的配置文件

3.7.4.1.用途

对于外部应用程序软件包的主机特定配置文件必须安装在 /etc/opt/<subdir> 中，这里 <subdir> 指 /opt 中保存该软件包的子树名称。

3.7.4.2. 要求

对 `/etc/opt/<subdir>` 中的内部分配方式没有结构上的要求。

如果为了使软件包或系统正常工作，某个配置文件必须旋转在另外的地方，可以将它放置在 `/etc/opt/<subdir>` 以外的地方。

> 基础知识
>
> 参见 `/opt` 的基础知识介绍。

3.7.5./etc/X11：X窗口系统的配置文件（可选）

3.7.5.1. 用途

`/etc/X11` 是放置所有X11主机特定配置的地方。为了允许在 `/usr` 以只读方式挂载的情况下进行本地控制，必需这个文件夹。

3.7.5.2.特殊选项

如果安装了相应子系统，`/etc/X11` 中必须有以下文件或链接：

文件	描述
Xconfig	早期XFree86版本的配置文件（可选）
XF86Config	XFree86第3版或第4版的配置文件（可选）
Xmodmap	全局X11键盘修改文件（可选）

`/etc/X11` 的子文件夹可以包含 `xdm` 的及其他程序（如一些窗口管理器）需要的文件[8]。

我们建议窗口管理器默认情况下只使用一个配置文件。`*wmrc` 文件必须命名为 `system.*wmrc` （除非另有一个被广泛接受的名称），并且不要使用子文件夹。任何窗口管理器的子文件夹必须根据实际的窗口管理器二进制文件清楚地命名。

3.7.6./etc/sgml：SGML的配置文件（可选）

3.7.6.1. 用途

定义了SGML系统高级参数的通用配置文件安装在这里。名称为 `*.conf` 的文件代表通用的配置文件。名为 `*.cat` 的文件是DTD特定的集中索引，包含了对所有其他使用给定DTD所需要的文件引用。超级索引文件 `catalog` 引用了所有集中索引文件。

3.7.7./etc/xml:XML的配置文件（可选）

3.7.7.1.用途

定义了XML系统高级参数的通用配置文件安装在这里。名称为 `*.conf` 的文件代表通用的配置文件。超级索引文件 `catalog` 引用了所有集中索引文件。

> 4. 启动时触发的命令脚本的设置可能会使用System V、BSD 或其他模式。标准以后可能会增加这部分的细节规范。↩

> 5. 我们建议将文件保存在/etc下的子文件夹中，而非直接保存在 `/etc` 下。↩

> 6. 使用影子密码机制的系统将在 `/etc` 下生成其他的配置文件（`/etc/shadow` 和其他文件）以及生成 `/usr/sbin` 下的其他程序（`usradd` 、`usrmod` 和其他程序）↩

> 7. 在一些Linux系统中，这可能是指向 `/proc/mounts` 的符号链接，此时不需要这种例外。↩

> 8. `/etc/X11/xdm` 保存了 `xdm` 的配置文件。它们多为以前 `/usr/lib/X11/xdm` 下的文件。一些 `xdm` 的本地变量数据保存在 `/var/lib/xdm` 中。↩

/home：用户主文件夹（可选）

3.8.1. 用途

`/home` 是一个相当标准的概念，但它显然是一个站点特定的文件系统[9]。对它的设置因主机而异。因而，程序不应该依赖这一位置[10]。

3.8.2. 要求

应用程序的用户特定配置保存在用户的主文件夹下一个以‘`.`’字符（“点文件”）开头形式的文件中。如果一个应用程序需要创建多于一个的点文件，它们应该放在一个以‘`.`’字符开头的子文件夹（点文件夹）中。这种情况下配置文件不应以‘`.`’开头[11]。

> [9]. 不同的人对用户账户放在哪里有不同的偏好。这一节只是描述了一种推荐的用户主文件夹位置；尽管如此，我们推荐所有与FHS兼容的发行版使用它作为主文件夹的默认位置。</br> 在小系统中，每位用户的文件夹典型情况下是 `/home` 下的子文件夹之一，如 `/home/smith` 、 `/home/torvalds` 、 `/home/operator` 等等。在大系统中（尤其是 `/home` 文件夹通过NFS在多个主机间共享时），有必要对用户主文件夹再细分。可以通过使用子文件夹进行细分，如 `/home/staff` （员工）、 `/home/guests` （访客）、 `/home/students` （学生）等等。↩

> [10]. 如果您想找到一位用户的主文件夹，您应该使用getwent（3）库函数，而不是依赖/etc/passwd文件，因为用户信息可能通过NIS这样的系统保存在远程位置。↩

> [11]. 建议程序在没有用户介入的情况下，除了自动保存和锁文件外，尽量避免在主文件夹下产生非点文件。↩

/lib：必需的共享库和内核模块

3.9.1. 用途

`/lib` 文件夹包含了启动系统和运行根文件系统中命令——如 `/bin` 和 `/sbin` 下的二进制文件——所需的共享库镜像[12]。

3.9.2. 要求

下列文件名模式至少要有一个（它们可以是文件或符号链接）：

文件	描述
libc.so.*	动态链接的C语言库（可选）
ld*	执行时链接器/装载器（可选）

如果安装了C预处理器，`/lib/cpp` 可以作为它的一个引用，这是出于历史原因[13]。

3.9.3. 特殊选项

如果安装了相应子系统，`/lib` 中应该有如下文件夹或符号链接：

文件夹	描述
modules	可加载内核模块（可选）

---

12. 只对 `/usr` 下的二进制文件（如任何的X窗口二进制文件）必需的共享库一定不要放在 `/lib` 下。只有运行 `/bin` 和 `/sbin` 下的二进制文件要求的共享库才可以放这里。特别的，`libm.so.*` 库如果没有任何 `/bin` 或 `/sbin` 下的程序需要它，也可以放在 `/usr/lib` 下面。↩

13. 这个二进制文件通常放在 `/usr/bin/cpp`。↩

/lib<qual>：替代格式的必需共享库（可选）

3.10.1. 用途

系统中可能有一个或多个 `/lib` 文件夹的变体，以支持多种二进制格式文件对各自库的要求[14]。

3.10.2. 要求

如果存在一个或多个这样的文件夹，对其内容的要求与正常的 `/lib` 文件夹相同，只是不要求有 `/lib<qual>/cpp`[15]。

> 14. 这经常在支持多种二进制格式文件而要求同名的库的系统中用于64位或32位支持。这种情况下，库文件夹可以是 `/lib32` 和 `/lib64`，`/lib` 作为其中之一的符号链接。↩

> 15. `/lib<qual>/cpp` 仍然允许使用：这样允许出现 `/lib` 和 `/lib<qual>` 相同的情况（一个是另一个的符号链接）。↩

/media：可移动媒质的挂载点

3.11.1. 用途

此文件夹包含了用作可移动媒质（如软盘、光盘和zip磁盘）挂载点的文件夹。

> 基础知识
>
> 历史上有多个不同的地方用来挂载可移动媒质，如`/cdrom`、`/mnt`或`/mnt/cdrom`。将可移动媒质的挂载点直接放置在根文件夹下可能导致/下大量的额外文件夹。尽管以`/mnt`下的文件夹作为挂载点近来变得普遍，这却与一个老得多的传统——直接使用`/mnt`作为临时挂载点——相冲突。

3.11.2. 特殊选项

如果安装了相应子系统，`/media`下必须有以下的文件夹或符号链接：

文件夹	描述
floppy	软盘驱动器（可选）
cdrom	CD-ROM驱动器（可选）
cdrecorder	CD刻录机（可选）
zip	zip驱动器（可选）

在存在多个挂载某种类型媒质的设备的系统中，创建挂载文件夹可以在这些可用设备名称后面加一个以‘`0`’开始的数字，但未计数的名称也应该保留[16]。

> 16. 照此规定，对于两个CDROM光驱，可以有`/media/cdrom0`和`/media/cdrom1`两个文件夹，`/media/cdrom`是其中之一的符号链接。↩

/mnt：临时挂载文件系统用的挂载点

3.12.1. 用途

提供这一目录是为了使系统管理员能够在需要时临时地挂载某个文件系统。此文件夹的内容是个因环境而异的问题，并且不应该到其中任何程序的运行方式。

此文件夹不应用于安装程序：可以转而使用另一个系统中未使用的合适的临时文件夹。

/opt：外部应用程序软件包

3.13.1. 用途

`/opt` 保留用于安装外部应用程序软件包。

要安装在 `/opt` 中的软件包必须将其静态文件放置在单独的一个 `/opt/<package>` 或 `/opt/<provider>` 文件夹树下，其中 `<package>` 是描述软件包的名称， `<provider>` 是软件提供者的Linux分配名称和数字授权（LANANA）的注册名称。

3.13.2. 要求

文件夹	描述
<package>	静态包对象
<provider>	LANADA注册的提供者名称

`/opt/bin` 、 `/opt/doc` 、 `/opt/include` 、 `/opt/info` 、 `/opt/lib` 和 `/opt/man` 保留给本地系统管理员使用。软件包可以由本地系统管理员在这些保留文件夹中提供“前台”文件（通过使用链接或复制），不过必须在脱离这些保留文件夹的情况下可以正常工作。

用户调用的程序必须位于 `/opt/<package>/bin` 文件夹或 `/opt/<provider>` 层次下。如果包中包含UNIX手册页面，它们必须位于 `/opt/<package>/share/man` 中或 `/opt/<provider>` 层次下，并且必须使用同 `/usr/share/man` 相同的子层次结构。

可变的包文件（在正常操作中会变化的文件）必须安装在 `/var/opt` 中。参见 `/var/opt` 了解更多信息。

主机特定的配置文件必须安装在 `/etc/opt` 中。参见 `/etc` 一节了解更多信息。

除了那些必须驻留在文件系统树中特定位置才能正常工作的包文件外，其他包文件不能出现在 `/opt` 、 `/var/opt` 和 `/etc/opt` 层次结构之外的地方。例外的情况比如，设备锁文件必须放置在 `/var/lock` 下，设备必须位于 `/dev` 中。

发行版可以将软件安装在 `/opt` 中，但未经本地系统管理员同意不能修改或删除本地系统管理员安装在该位置的软件。

基础知识

使用 `/opt` 安装外部软件是UNIX社区中早已建立的惯例。基于System V接口定义（第三版）的System V应用程序二进制文件接口[AT&T 1990],也提供了一个与这里定义十分相似的 `/opt` 结构。

Intel二进制兼容标准v.2（iBCS2）也为 `/opt` 提供了一个类似的结构。

通常情况下，支持系统中某个软件包的所有数据都应该放在 `/opt/<package>` 里，包括将要复制到 `/etc/opt/<package>` 和 `/var/opt/<package>` 中的文件，和保留在 `/opt` 中的文件夹一样。

对使用 `/opt` 的发行版所做的最小限制是必要的，因为在发行版安装的软件和本地安装的软件之间可能存在冲突，尤其是在某些二进制文件中出现固定路径名称的情况下。

`/opt/<provider>` 下的文件夹结构留给软件打包者自己规定，尽管我们建议软件包安装在 `/opt/<provider>/<package>` 中，并遵循 `/opt/package` 的指导原则。从这种结构可以合理地引申出支持文件可以安装在 `/opt<provider>/lib` 或 `/opt/<provider>/bin` 中。

/root：root用户的主文件夹（可选）

3.14.1. 用途

root 账户的主文件夹可以由开发者或本地偏好设置来决定，但这是推荐的默认位置[17]。

> 17. 如果root账户的主文件夹不在根分区外 ，需要在无法找到它的情况下将此文件夹默认定位到 `/` 。</br> 我们不推荐使用root账户进行未授权用户可以完成的任务，而只用它来进行系统管理。出于此原因，我们推荐root账户的主文件夹下不要出现邮件和其他应用的子文件夹，给管理者如**root**、**postmaster**和**webmaster**等的信件转发给某个合适的（一般）用户。↩

/sbin：系统二进制文件

3.15.1. 用途

系统管理员使用的工具（和其他只有root使用的命令）保存在 `/sbin`、`/usr/sbin` 和 `/usr/local/sbi` n中。`/sbin` 包含除了 `/bin` 中之外其他的启动、还原、恢复和/或修复系统所必需的二进制文件[18]。

在已知 `/usr` 已经挂载（并且没有问题）的情况下执行的程序通常位于 `/usr/sbin` 下。本地安装的系统管理程序应该位于 `/usr/local/sbin` 下[19]。

3.15.2. 要求

`/sbin` 下应该有以下命令或符号链接。

命令	描述
shutdown	关闭系统的命令

3.15.3. 特殊选项

如果安装了相应子系统，`/sbin` 下必须有以下文件或符号链接：

命令	描述
fastboot	不检查磁盘重启系统（可选）
fasthalt	不检查磁盘停止系统（可选）
fdisk	操作分区表（可选）
fsck	文件系统检查和修复工具（可选）
fsck.*	针对某一特定文件系统的检查和修复工具（可选）
getty	`getty` 程序（可选）
halt	停止系统的命令（可选）
ifconfig	配置网络接口的命令（可选）
init	初始化进程（可选）
mkfs	创建文件系统的命令（可选）
mkfs.*	创建特定文件系统的命令（可选）
mkswap	设置交换区的命令（可选）
reboot	重启系统的命令（可选）
route	IP路由表工具（可选）
swapon	启用分页和交换（可选）
swapoff	关闭分页和交换（可选）
update	周期性的清洗文件系统缓存区的后台服务（可选）

---

18. 原来，/sbin二进制文件保存在/etc下。↩

19. 决定哪些东西应该放在“`sbin`”中很简单：如果一个正常（非系统管理员）用户有时会直接运行它，那它必须放在某一“`bin`”文件夹中。普通用户的路径中应该不必出现`sbin`文件夹。</br>例如，像chfn这样用户只是使用的文件仍然必须放在`/usr/bin`中。`ping`这个命令尽管对于`root`是绝对必须的（网络还原和诊断），却经常被用户使用，所以必须驻留在`/bin`下。</br>我们推荐用户对于`/sbin`下的所有东西有读和执行的权限，除了一些可能进行了`setuid`和`setgid`设置的程序。区分`/bin`和`/sbin`不是为了安全原因或阻止用户查看操作系统，而是为了清晰地区别每个人都能用的和最初用于系统管理任务的二进制文件。把用户拦在`/sbin`外也起不到实质的提升安全性的作用。↩

/srv：本系统所提供服务的数据文件

3.16.1. 用途

`/srv` 包含了由系统提供的站点特定的数据。

> 基础知识
>
> 规定这一目录的目的是使用户可以找到特定服务的数据文件的位置，因而要求单独的树来存放只读数据、可写数据和脚本（如cgi脚本）的服务可以放这里。只与某位用户相关的数据应该放入该用户的主文件夹下。
>
> `/srv` 下子文件夹的命名方法尚未规定，因为目前还没有就这方面规定的一致意见。对 `/srv` 下数据进行归类的一个办法是按协议，如ftp、rsync、www和cvs。在大型系统中，用管理内容来归类会有好处，如 `/srv/physics/www` 、 `/srv/compsci/cvs` 等等。设置在每个主机上都可能不同。因而，任何程序都不应依赖于/srv下现有的特定子文件夹或必要保存在 `/srv` 下的数据。然而，FHS兼容系统中总应存在 `/srv` 文件夹并且作为这些数据的默认位置。
>
> 发行版必须必须注意不要在未经管理员允许的情况下删除这些文件夹中的文件[20]。

---

20. 这特别重要，因为这些地方经常同时包含发行者最初安装的和管理员添加的文件。↩

/tmp：临时文件

3.17.1. 用途

`/tmp` 文件夹必须能够为要求临时文件的程序使用。

程序必须假设 `/tmp` 中的任何文件或文件夹仅用于在程序中引用。

> ### 基础知识
>
> IEEE标准 P1003.2（POSIX，第2部分）做出了与上节相似的要求。
>
> 尽管 `/tmp` 中的文件可以站点指定的方式保存，我们依然建议在每次系统启动时删除 `/tmp` 下的文件和文件夹。
>
> FHS基于历史先例和习惯做法增加了这条建议，但不作为要求，因为系统管理不在此标准讨论的范围内。

/usr 层次结构

用途

`/usr` 是文件系统中的第二个重要的部分。 `/usr` 是可共享的只读数据。就是说 `/usr` 应该可以在各种FHS兼容的主机之间共享并且禁止写入。任何主机特有的信息或随时间变化的量都保存在其他地方。 大型的软件包禁止在 `/usr` 层次结构中使用直接的子文件夹。

要求

`/usr` 中应该有以下文件夹或符号链接。

文件夹	描述
bin	绝大多数用户命令
include	C程序包含的头文件
lib	库
local	本地层次结构（整个系统安装后为空）
sbin	非紧要的系统二进制文件
share	独立与系统架构的数据

特殊选项

文件夹	描述
X11R6	X窗口系统，版本11发行号6（可选）
hames	游戏和教育二进制文件（可选）
lib	替换格式的库（可选）
src	源代码（可选）

为X窗口系统留出一个例外是因为有众多的先例并且在实践中被广泛认可。

可以有到以下到文件夹的符号链接。留下这种可能性是为了满足保持与旧系统兼容性的需要，直到可以假定所有实现都使用/var文件夹为止（再禁止使用）。

- `/usr/spool` -> `/var/spool`
- `/usr/tmp` -> `/var/tmp`
- `/usr/spool/locks` -> `/var/lock`

一旦系统不再需要以上任何一个链接，可以在需要时删除它。

/usr/X11R6：X窗口系统，版本11发行号6（可选）

4.4.1. 用途

这一层次结构保留给X窗口系统·版本11发行号6和相关文件。

为了简化并使XFree96与其他系统上的X窗口系统更兼容，如果存在 `/usr/X11R6` ，以下符号链接必须存在：

- `/usr/bin/X11` -> `/usr/X11R6/bin`
- `/usr/lib/X11` -> `/usr/X11R6/lib/X11`
- `/usr/include/X11` -> `/usr/X11R6/include/X11`

通常，软件不得使用以上的符号链接安装或管理。它们只为用户使用而设。引起的困难是X窗口系统的发行版本问题——在过渡期，无法知道正在使用的X11是什么发行版本。

4.4.2. 特殊选项

`/usr/X11R6/lib/X11` 下的主机特定数据应该解释为演示文件。要求当前主机信息的应用程序应该参考 `/etc/X11` 中的配置文件（来查找），这个配置文件可以是指向 `/usr/X11R6/lib` 中某个文件的链接[1]。

---

1. 这样的配置文件的例子包括Xconfig、XF86Config，或system.twmrc。↩

/usr/bin：绝大多数用户命令

4.5.1. 特殊选项

如果安装了相应子系统，`/usr/bin`下应该有如下文件夹或符号链接：

文件夹	描述
mh	MH邮件处理系统的命令（可选）

如果`/usr/X11R6/bin`存在，`/usr/bin/X11`必须是指向它的链接。

如果安装了相应子系统，`/usr/bin`中必须存在以下文件或符号链接：

命令	描述
perl	实用提取和报告语言（Perl，可选）
python	Python解释型语言（可选）
tclsh	包含Tcl解释器的简单shell（可选）
wish	简单的Tcl/Tk窗口shell（可选）
expect	交互式对话框程序（可选）

> 基础知识
>
> 因为shell脚本解释器(以shell脚本第一行的`#!<path>`触发)不能依赖于路径，对它们的位置进行标准化是有好处的。Bourne shell和C-shell解释器已经固定于`/bin`中，但`Perl`、`Python`和`Tcl`常常在许多不同的地方出现。以上这些可以是指向shell解释器的实际位置的符号链接。

/usr/include：标准头文件文件夹

4.6.1. 用途

这是应放置所有系统通用的C程序设计语言头文件的地方。

4.6.2. 特殊选项

如果安装了相应子系统，`/usr/include` 下必须有如下文件夹或符号链接：

文件夹 |描述 bsd |BSD兼容头文件（可选）

如果 `/usr/X11R6/include/X11` 存在，则符号链接 `/usr/include/X11` 必须指向它。

/usr/lib：编程和软件包的库

4.7.1. 用途

`/usr/lib` 包含用于用户或shell脚本直接调用的目标文件、库文件和内部二进制文件[2]。

应用程序可以在 `/usr/lib` 下有一个单独的子文件夹。如果一个应用程序使用了一个子文件夹，所有该应用程序专用的依赖于架构的数据必须也位于此子文件夹中[3]。

4.7.2. 特殊选项

由于历史原因，如果 `/usr/sbin/sendmail` 存在，则 `/usr/lib/sendmail` 必须是指向它的符号链接[4]。

如果 `/lib/X11` 存在，`/usr/lib/X11` 必须是一个指向它的符号链接，或（若 `/usr/lib` 也是符号链接）指向 `/lib/X11` 指向的任何对象[5]。

> 2. 各种独立于架构的应用程序特定的静态文件和子文件夹必须放置在/usr/share中。↩

> 3. 例如，`Perl5` 模块和库的 `perl5` 子文件夹。↩

> 4. 一些可执行命令如 `makewhatis` 和 `sendmail` 传统上也放在 `/usr/lib` 中。`Makewhatis` 是一个内部的二进制文件，必须放在二进制文件夹中；用户只访问 `catman`。新的 `sendmail` 二进制文件现在默认放在 `/usr/sbin` 中。另外，使用与 `sendmail` 兼容的邮件传输代理的系统必须提供 `/usr/sbin/sendmail` 这个到适当可执行文件的符号链接。↩

> 5. X窗口系统的主机特定数据禁止保存在 `/usr/lib/X11`。主机特定的配置文件如 `Xconfig` 或 `XF86Config` 必须保存在 `/etc/X11` 中。中包括像 `system.Twmrc` 这样的配置数据，即使它只是一个到更全局配置文件（可能在 `/usr/X11R6/lib/X11` 中）的符号链接。↩

/usr/lib<qual>：替换格式的库（可选）

4.8.1. 用途

`/usr/lib<qual>` 用于替换的二进制格式，与 `/usr/lib` 扮演相同的角色，只是不要求有 `/usr/lib<qual>/sendmal` 和 `/usr/lib<qual>/X11` 两个符号链接[6]。

4.8.2. /usr/local：本地层次结构

4.8.2.1 用途

`/usr/local` 层次结构用于系统管理员安装本地软件。在系统软件升级时它要保证安全，不被覆盖。它可以用作放置可以在一组主机之间共享而在/usr中找不到的程序和数据。

本地安装的软件必须位于 `/usr/local` 中，而不是 `/usr` ，除非想安装它来替换或升级 `/usr` 中的软件[7]。

4.8.2.2. 要求

`/usr/local` 下必须有以下文件夹或符号链接：

文件夹	描述
bin	本地二进制文件
btc	本地二进制文件的主机特定系统配置
bames	本地游戏二进制文件
include	本地C头文件
lib	本地库
man	本地在线手册
sbin	本地系统二进制文件
share	本地的与架构独立的层次结构
src	本地源代码

除了下面列出的文件夹外，在安装FHS兼容的系统之后， `/usr/local` 下不应该有其他文件夹。

4.8.2.3. 特殊选项

如果 `/lib<qual>` 或 `/usr/lib<qual>` 存在， `/usr/local` 下必须有对等的文件夹。

`/usr/local/etc` 可以是指向 `/etc/local` 的符号链接。

> ### 基础知识
>
> 保持 `/usr/local/etc` 的一致性对安装者有利，并且已经在其他系统中使用。在需要备份所有 `/usr/local` 下的所有东西来再创建一个系统的情况下，这样做不会引入其他的维护开销。但如果系统想把所有它们的配置放到同一层次结构中，建一个指向/etc/local的符号链接比较合适。
>
> 注意 `/usr/etc` 仍然不允许使用： `/usr` 中的程序应该将配置文件放在 `/etc` 下。

> 6. 如果/usr/lib和/usr/lib相同（一个是另一个的符号链接），将存在这些文件和每个应用程序的子文件夹。↩

> 7. 位于 `/` 和 `/usr` 中的软件可能被系统升级覆盖（尽管我们建议这些情况下发行版不要覆盖 `/etc` 中的数据）。因此，没有充分的理由本地软件不要放置在 `/usr/local` 以外的地方。↩

/usr/local/share

对此文件夹下内容的要求同 `/usr/share` 。唯一追加的限定是 `/usr/local/share/man` 和 `/usr/local/man` 文件夹必须同步（通常意味着其中之一为符号链接）[8]。

> 8. `/usr/local/man` 可能在将来的FHS版本中废弃，因此如果东西放哪都一样，将那个（`/usr/local/man`）做成符号链接比较明智。↩

/usr/sbin：非关键的标准系统二进制文件

4.10.1. 用途

此文件夹包含了任何系统管理员单独使用的任何非关键二进制文件。系统修复、还原、挂载`/usr`或其他关键功能的系统管理程序必须转而放在/sbin下[9]。

> 9. 本地安装的系统管理程序应该位于`/usr/local/sbin`。↩

4.11. /usr/share：与架构独立的数据

4.11.1. 用途

`/usr/share` 层次结构用于所有只读的与架构独立的数据文件[10]。

这一层次结构应该可以在给定操作系统的所有架构平台之间共享；因而，例如对i386、Alpha和PPC平台的站点可以维护一个集中挂载的 `/usr/share` 文件夹。不过注意，一般不应该在不同的操作系统间或相同操作系统的不同版本间共享 `/usr/share` 。

任何包含或要求不需要修改数据的程序或软件包应该将这些数据保存在 `/usr/share` （或 `/usr/local/share` ，本地安装时）下。建议使用 `/usr/share` 下的一个子文件夹来放置它们。

在 `/usr/share/games` 下的游戏数据应该是纯静态的数据。任何可修改的文件如分数文件、游戏日志之类，应该放置在 `/var/games` 中。

4.11.2. 要求

`/usr/share` 下必须有如下文件夹或符号链接：

文件夹	描述
man	在线手册
misc	各种与架构独立的数据

4.11.3. 特殊选项

如果安装了相应子系统，则 `/usr/share` 中必须有以下文件夹或符号链接：

文件夹	描述
dict	单词列表（可选）
doc	各种文档（可选）
games	`/usr/games` 的静态数据文件（可选）
info	GNU Info系统的主文件夹（可选）
locale	区域信息（可选）
nls	本地语言支持的消息索引（可选）
sgml	SGML数据（可选）
terminfo	Terminfo数据库的文件夹（可选）
tmac	未使用 `groff` 发布的 `troff` 宏（可选）
xml	XML数据（可选）
zoneinfo	时区信息和配置（可选）

建议程序特定的、与架构独立的文件夹放在这里。这些文件夹包括 `groff` 、 `perl` 、 `ghostscript` 、 `texmf` 和 `kbd` （Linux）或 `syscons` （BSD）。然而也可以将它们放在 `/usr/lib` 中以保持向后兼容，这由发行者确定。类似的，如果发行者想要放置游戏数据，也可以在 `/usr/share/games` 层次结构之外使用 `/usr/lib/games` 层次结构。

4.11.4. /usr/share/dict：单词表（可选）

4.11.4.1. 用途

此文件夹是放系统单词表的地方。传统上这一文件夹只包含英语words文件，它用于 `look` (1) 和各种拼写程序。Words可以使用美式或英式拼写。

> 基础知识
>
> 只有单词表放在此处的原因是它们是所有拼写检查程序之间惟一通用的文件。

4.11.4.2. 特殊选项

如果安装了相应子系统， `/usr/share/dict` 下必须有以下文件或符号链接：

文件	描述
words	英语单词表（可选）

同时需要英式和美式拼写的站点可以将 `words` 链接到 `/usr/share/dict/american-english` 或 `/usr/share/dict/british-english` 。

其他语言的单词表可以使用该语言的英文名称添加，例如，`/usr/share/dict/french`、`/usr/share/dict/danish`，等等。对有问题的语言，可能的话应该使用合适的ISO 8859字符集；如果可能，应该使用Latin1(ISO 8859-1)字符集（通常不适用）。

如果存在，这里可以包含其他单词表。

4.11.5. /usr/share/man：手册页

4.11.5.1. 用途

本节细述整个系统中手册页的组织，包括`/usr/share/man`。也提到了`/var/cache/man`那一节。

系统中`<mandir>`的初始位置是`/usr/share/man`。`/usr/share/man`包含了`/`和`/usr`文件系统下命令和数据的手册信息[11]。

手册页保存在`<mandir>/<locale>/man<section>/<arch>`中。对`<mandir>`、`<locale>`、`<section>`和`<arch>`的解释在下面给出。

下面是手册各节的描述：

- Man1：用户程序手册。本节中包含了描述公共访问的命令的手册页面。用户需要使用的绝大多数程序文档位于这里。
- Man2：系统调用。这一节描述了所有的系统调用（对内核执行操作的请求）。
- Man3：库函数和子例程。第3节描述了没有直接调用系统服务的程序库例程。这一节与第2节实际上只针对编辑者。
- Man4：特殊文件。第4节描述了系统中的特殊文件、有关的驱动功能和网络支持。典型的，这包括/dev下能找到的设备文件和内核对网络协议的支持接口。
- Man5：文件格式。很多数据文件格式的文档都在第5节。这包括各种头文件、程序输出文件和系统文件。
- Man6：游戏。这一节是游戏、演示和通常很琐碎的程序的文档。对这些内容的必要性，不同的人有不同的见解。
- Man7：各种难以归类的手册页划归第7节。Troff和其他文本处理宏包（的手册）放在这里。
- Man8：系统管理员用来进行系统操作和维护的系统管理程序的文档放在这里。有些程序有时也对普通用户有用。

4.11.5.2. 特殊选项

`/usr/share/<mandir>/<locale>`下必须有以下文件夹或符号链接，除非它们是空的[12]：

文件夹	描述
man1	用户程序（可选）
man2	系统调用（可选）
man3	库调用（可选）
man4	特殊文件（可选）
man5	文件格式（可选）
man6	游戏（可选）
man7	杂项
man8	系统管理（可选）

`<section>` 部分描述了手册的节次。

必须在 `/usr/share/man` 下预设（子文件夹）来支持各种不同（或多种）语言的手册文件。这些预设必须考虑存储和手册间的相互引用。有关因素包括语言（包括地域差异）和字符编码。

`/usr/share/man` 下语言子文件夹的命名基于附录E中的POSIX 1003.1标准，它描述了区域识别字符串——描述文化环境最被广泛接受的方法。 `<locale>` 字符串是： `<language>[_<territory>][.<character-set>][,<version>]`

`<language>` 字段必须取自ISO 639（代表语言名的一套代码）。它必须是两个字符宽且必须只能为小写。

`<territory>` 字段必须是ISO 3166（代表国家的一种规定）中的双字符代码，如果里面有。（绝大多数人应该对电子邮件中使用的双字符国家代码比较熟悉。）它必须是两个字符宽，且只能为大写[13]。

可以在 `<character-set>` 字段后放一个指定 `<version>` （版本）的参数，用逗号分开。这可以用来区别各种不同的文化需要；例如，词典顺序对比一个更面向系统的整理顺序。本标准不推荐使用 `<version>` 字段，除非需要。

手册页使用单一语言和代码集的系统可以忽略 `<locale>` 子字符串并把所有手册页保存在 `<mandir>` 下。例如，只有以ASCII编码的英文手册页的系统可以将手册页（ `man<section>` 文件夹）直接保存在 `/usr/share/man` 下。（传统的环境中实际上就是这么安排的。）

对于有一套广泛接受字符编码集的国家，可以忽略 `<character-set>` 字段，但是强烈建议保存它，尤其是对于有数套相互冲突标准的国家。

各种例子：

语言 Language	地区 Territory	字符集Character set	文件夹
英语	——	ASCII	/usr/share/man/en
英语	英国	ISO 8859-15	/usr/share/man/en_GB
英语	美国	ASCII	/usr/share/man/en_US
法语	加拿大	ISO 8859-1	/usr/share/man/fr_CA
法语	法国	ISO 8859-1	/usr/share/man/fr_FR
德语	德国	ISO 646	/usr/share/man/de_DE.646
德语	德国	ISO 6937	/usr/share/man/de_DE.6937
德语	德国	ISO 8859-1	/usr/share/man/de_DE.88591
德语	瑞士	ISO 646	/usr/share/man/de_CH.646
日语	日本	JIS	/usr/share/man/ja_JP.jis
日语	日本	SJIS	/usr/share/man/ja_JP.sjis
日语	日本	UJIS(或EUC-J)	/usr/share/man/ja_JP.ujis
汉语	中国大陆	UTF-8	/usr/share/man/zh_CN
汉语	中国台湾	UTF-8	/usr/share/man/zh_TW

> 译注：汉语是我加的。个人建议不要使用中文字符集，统一使用UTF-8。

类似的，为手册页所做的预设必须是独立于架构的，如设备驱动或系统管理命令的文档。这些必须安置在相应 `man<section>` 文件夹下的某个 `<arch>` 文件夹中；例如，i386架构的 `ctrlaltdel` (8)命令的man页面可以放在 `/usr/share/man/<locale>/man8/i386/ctrlaltdel.8` 。

`/usr/local` 下命令和数据的手册页保存在 `/usr/local/man` 下。X11R6的手册页保存在 `/usr/X11R6/man` 下。同理，系统中所有的手册页必须与 `/usr/share/man` 保持相同的结构。

包含格式化的手册页记录的cat页面的各节（ `cat<section>` ）也应位于 `<mandir/<locale>` 下，但不要求并且也不允许以nroff source手册页的lieu形式发布。

各节从“1”到“8”加序号的做法是按传统定义的。通常，位于特定节中的手册页文件名以 `.<section>` 结尾。

另外，一些应用程序特定的大型手册页文件名带有附加的后缀。MH邮件处理系统手册页必须在所有MH手册后加 `mh` 后缀。所有X窗口系统的手册页文件名后必须附加一个 `x` 。

`/usr/share/man` 下存放各种语言的手册页到合适的子文件夹的做法也适用于其他的手册页层次结构，如 `/usr/local/man` 和 `/usr/X11R6/man` 。（这部分的标准也适用于后面可选的 `/var/cache/man` 结构那节。）

4.11.6. /usr/share/misc：各种独立于架构的数据

此文件夹包含了各种独立于架构的、不需要在/usr/share下单独开辟子文件夹的文件。

4.11.6.1. 特殊选项

如果安装了相应子系统，则 `/usr/share/misc` 下必须有以下文件或符号链接：

文件	描述
ascii	ASCII字符集表（可选）
magic	文件命令用的魔数的默认列表（可选）
termcap	终端功能数据库（可选）
termcap.db	终端功能数据库（可选）

其他（应用程序特定的）文件也可以放在这里，但发行者可以酌情将它们放在 `/usr/lib` /中[14]。

4.11.7. /usr/share/sgml：SGML数据（可选）

4.11.7.1. 用途

`/usr/share/sgml` 包含了SGML应用程序使用的独立于架构的文件，如普通索引（不是集中索引，参见 `/etc/sgml` ），DTD、实体或样式表。

4.11.7.2. 特殊选项

如果安装了相应子系统，则 `/usr/share/sgml` 中必须有以下文件夹或符号链接：

文件夹	描述
docbook	Docbook DTD（可选）
tei Tei	DTD（可选）
html	Html DTD（可选）
mathml	Mathml DTD（可选）

没有专门对应于给定DTD的其他文件可以放在它们自己的子文件夹中。

4.11.8. /usr/share/xml：XML数据（可选）

4.11.8.1. 用途

`/usr/share/xml` 包含XML应用程序使用的独立于架构的文件，如普通索引（不是集中索引，参见 `/etc/sgml` ），DTD、实体或样式表。

4.11.8.2. 特殊选项

如果安装了相应子系统，`/usr/share/xml` 下必须有以下文件夹或符号链接：

文件夹	描述
docbook	Docbook XML DTD（可选）
xhtml	XHTML DTD（可选）
mathml	MathML DTD（可选）

---

10. 这种数据以前有很多存放在/usr（man、doc）或/usr/lib（dict、terminfo、zoneinfo）。↩

11. 显然，`/` 下没有手册文件，因为启动时不需要，而且实际上紧急情况下也用不着。↩

12. 例如，如果 `/usr/local/man` 第4节（设备）中没有手册页，那么 `/usr/local/man/man4` 可以忽略。↩

13. 很突出的一个例外就是对于英国的规则，在ISO 3166中是‘GB’，但绝大多数电子邮件地址中是‘UK’。↩

14. 例如：`airport` 、`birthtoken` 、`eqnchar` 、`getopt` 、`gprof.callg` 、`gprof.flat` 、`inter.phone` 、`ipfw.samp` . `filters` 、`ipfw.samp.scripts` 、`keycap.pcvt` 、`mail.help` , `mail.tildehelp` 、`man.template` 、`map3270` 、`mdoc.template` 、`more.help` 、`na.Phone` 、`nslookup.help` 、`operator` 、`scsi_modes` 、`sendmail.hf` 、`style` 、`units.lib` 、`vgrindefs` 、`vgrindefs.db` 、`zipcodes` 等文件。↩

/usr/src：源代码（可选）

4.12.1. 用途

源代码可以放在这个子文件夹中，仅用作参考[12]。

> 12. 通常此层次结构下不应该进行源文件编译。↩

/var层次结构

用途

`/var` 包含可变数据文件。这包括 `spool` 文件夹和文件、管理和日志数据，及过渡和临时文件。

`/var` 中有一些部分不能在不同的系统间共享。例如， `/var/log` 、 `/var/log` 和 `/var/run` 。另外的部分可以共享，尤其像 `/var/mail` 、 `/var/mail` 、 `/var/cache/man` 、 `/var/cache/fonts` 和 `/var/spool/news` 。

在这里对 `/var` 作出详细规定的目的是可以用只读方式挂载 `/usr` 。想放在 `/usr` 中的东西只要可能在操纵系统（为了安装和软件维护）时写入内容，就必须将它们放在 `/var` 中。

如果 `/var` 不能放在一个单独的分区上，也最好将它移出根分区，放在 `/usr` 所在的分区上。（这样做有时是为了减少根分区的大小或防止根分区空间变得过少。）然而，一定不要将 `/var` 链接到 `/usr` ，这样会使区分 `/usr` 和 `/var` 更加困难，并且可能造成命名冲突。把 `/var` 链接到 `/usr/var` 。

通常禁止应用程序在 `/var` 顶层中直接添加文件夹。只有在对整个系统有一定意义的情况下才应该创建这样的文件夹，并且要先向FHS邮件列表咨询。

要求

`/var` 下要求有这样的文件夹或符号链接。

文件夹	描述
cache	应用程序缓存数据
lib	可变状态信息
local	`/usr/local` 的可变数据
lock	锁文件
log	日志文件和文件夹
opt	`/opt` 的可变数据
run	有关正在运行进程的数据
spool	应用程序spool数据
tmp	系统两次启动之间保留的临时文件

有几个文件夹是'保留的'，就是说，轻易不许一些新的应用程序使用它们，有可能与旧的或系统中的用法冲突。它们是：

- /var/backups
- /var/cron
- /var/msgs
- /var/preserve

特殊选项

如果安装了相应子系统，`/var` 中必须有以下文件夹或符号链接：

文件夹 |描述 account |进程账户记录（可选） crash |系统崩溃的转储数据（可选） games |可变的游戏数据（可选） mail |用户邮箱文件（可选） yp |网络信息服务（NIS）数据库文件（可选）

/var/account：进程账户记录（可选）

5.4.1. 用途

这一文件夹保存当前活动进程的账户记录和进程树的使用数据（类UNIX系统中的 `lastcomm` 和 `sa` 会使用）。

/var/cache：应用程序缓存数据

5.5.1. 用途

`/var/cache` 用于放置来自应用程序的缓存数据。这些数据是本地计算机上耗时的I/O操作或计算生成的。应用程序必须能够重新生成或还原这些数据。与 `/var/spool` 不同，应时的缓存文件可以删除，不会有数据损失。在两次调用该应用程序或重启系统间这些数据必须一直有效。

位于 `/var/cache` 下的文件可能在应用程序的某种特殊行为中、或被系统管理员标记为过期，或者兼有这两种情况。该应用程序必须总是有能力恢复被手动删除（通常是因为磁盘空间不足）的这些文件。对这些缓存文件夹的数据格式没有其他要求。

> 基础知识
>
> 留出一个单独的文件夹给缓存数据，系统管理员就可以对它采取与/var其他文件夹不同的磁盘和备份策略。

5.5.2. 特殊选项

文件夹	描述
fonts	本地生成的字体（可选）
man	本地格式化的手册页（可选）
www	WWW代理或缓存数据（可选）
<package>	包特定的缓存数据（可选）

5.5.3. /var/cache/fonts：本地生成的字体（可选）

5.5.3.1. 用途

`/var/cache/fonts` 应该用来保存任何动态创建的字体。尤其是所有 `mktexpk` 自动生成的字体必须放在 `/var/cache/fonts` 下相应名称的子文件夹中[1]。

5.5.3.2. 特殊选项

其他动态创建的字体也应该放在此文件夹树中、`/var/cache/fonts` 中相应名称的子文件夹下。

5.5.4. /var/cache/man：本地格式化的手册页（可选）

5.5.4.1. 用途

此文件夹为使用只读 `/usr` 分区、而想允许缓存本地格式化的手册页的站点提供了一个标准位置。以可写方式挂载 `/usr` 的站点（如单用户的系统）可以不使用 `/var/cache/man`，而是将格式化的手册页直接写入 `/usr/share/man` 下的 `cat<section>` 文件夹中。我们建议多数站点转而使用以下选项之一：

- 将所有手册页与未格式化的手册页一起格式化。
- 不允许缓存格式化手册页，并要求每次提供手册页时必须完成格式化。
- 允许在/var/cache/man中本地缓存格式化的手册页。

`/var/cache/man` 的结构需要既要反映多种手册页层次结构的状况，也要反映支持多种语言的可能性。 如果未格式化的手册页正常情况下出现在 `<path>/man/<locale>/man<section>` 中，则放置格式化的手册页的文件夹是 `/var/cache/man<catpath>/locale>/cat<section>` ，其中 `<catpath>` 是从 `<path>` 中移除路径名中任何前导的 `/usr` 和 `/` 或尾随的 `share` 部分得到的[2]。（注意可能丢失部分。）

写入 `/var/cache/man` 中的手册页最终可能被转移到源 `man` 层次结构中相应的预格式化文件夹中，或者丢弃；类似的，在源 `man` 层次结构中的格式化手册页也可能在一段时间不访问后作废。

如果预格式化的手册页是在只读媒质上带有的（如CD-ROM上），必须将它们安装到源 `man` 层次结构中（如 `/usr/share/man/cat<sections>` ）。 `/var/cache/man` 是预留作可写的格式化手册页缓存区的。

> 基础知识
>
> 本标准的1.2版为此层次结构规定了 `/var/catman` 路径。把该路径移到 `/var/cache` 是为了更好地反映格式化手册页的动态属性。文件夹名改成了 `man` ，以改进该层次结构来容纳“cat”之外的其他处理过的格式，如PostScript、HTML或DVI。

---

1. 本标准目前没有吸纳TeX文件夹结构（一篇描述TeX文件和文件夹布局的文档），但读读它也许有用。它位于ftp:/ctan.tug.org/tex/。↩

2. 如， `/usr/share/man/man1/ls.1` 格式化为 `/var/cache/man/cat1/ls.1` ，而 `/usr/X11R6/man/<locale>/man3/XtClass.3x` 格式化为 `/var/cache/man/X11R6/<locale>/cat3/XtClass.3x` 。↩

/var/crash：系统崩溃转储（可选）

5.6.1. 用途

此文件夹保存系统崩溃转储。此标准发布时，Linux还不支持系统崩溃转储，但其他系统可能支持，这样就可以与FHS兼容。

/var/games：各种游戏数据（可选）

5.7.1. 用途

与 `/usr` 中游戏相关的任何变量数据应该放置在这里。 `/var/games` 应该存放原先在 `/usr` 中的可变数据，帮助文本、等级描述等静态数据必须在其他地方保存，如 `/usr/share/games` 。

> 基础知识
>
> `/var/games` 被分配了一个单独的层次结构，而不是照标准1.2版那样与旧的 `/var/lib` 合并在一起。这样划分可以方便对备份策略、权限和磁盘使用的本地控制，以及允许主机间共享并减少 `/var/lib` 中的簇数量。另外， `/var/games` 是BSD系统一贯使用的路径。

/var/lib：可变状态信息

5.8.1. 用途

这一层次结构保存属于某个应用程序或系统的状态信息。状态信息是程序运行时会修改的和属于某个特定主机的信息。禁止用户修改 `/var/lib` 中的信息来配置软件包的操作方式。

状态信息通常用来保持一个（或一组内部相关的）应用程序多次调用和不同实例之间的状态。通常重启后状态信息应该依然有效，不应该是日志输出，并且不应该是spool数据。

一个应用程序（或一组内部相关的应用程序）必须使用 `/var/lib` 下的一个子文件夹来保存其数据。规定了一个 `/var/lib/misc` 文件夹用作保存不需要子文件夹的状态文件；其他子文件夹只有该发行版中有相关的应用程序时才可以出现[3]。

`/var/lib/<name>` 是必须用于所有发行版打包支持的位置。当然，不同的发行版可以用不同的位置。

5.8.2 要求

`/var/lib` 下要求有如下文件夹或符号链接：

文件夹	描述
misc	混杂的状态数据

5.8.3 特殊选项

如果安装了相应子系统，`/var/lib` 下必须有以下文件夹或符号链接：

文件夹	描述
<editor>	编辑器备份文件和状态（可选）
<pkgtool>	打包支持文件（可选）
<package>	包和子系统的状态数据（可选）
hwclock	Hwclock的状态文件夹（可选）
Xdm	X显示管理器可变数据（可选）

5.8.4. /var/lib/：编辑器备份文件和状态（可选）

5.8.4.1. 用途

这些文件夹保存了任何意外终止的编辑器（如 `elvis` 、 `jove` 、 `nvi` ）所产生的文件。其他编辑器可能不需要一个崩溃恢复文件的文件夹，但需要一个定义好的地方来保存运行时的其他信息。这类信息应该保存在 `/var/lib` 下的子文件夹中（如GNU Emacs应将锁文件放在 `/var/lib/emacs/lock` 中）。

未来的编辑器可能需要崩溃恢复文件和锁文件之外的其他状态信息，那类信息也应该放置在 `/var/lib/<editor>` 下。

> 基础知识
>
> 以前的Linux发行版及所有商业厂商使用 `/var/preserve` 来保存 `vi` 及其衍生版的状态文件。然而，每个编辑器使用自有格式的崩溃-恢复文件，因而也各自需要一个单独的文件夹。
>
> 编辑器特定的锁文件通常与保存在 `/var/lock` 中的设备或资源锁文件有很大不同，它们保存在 `/var/lib` 。

5.8.5. /var/lib/hwclock：hwclock的状态文件夹（可选）

5.8.5.1. 用途

这一文件夹存放 `/var/lib/hwclock/adjtime` 文件。

> 基础知识
>
> 在FHS2.1中，该文件为 `/etc/adjtime` ，但 `hwclock` 程序对此作了发动，原来的文件已不适用。

5.8.6. /var/lib/misc：杂类变量数据

5.8.6.1. 用途

此文件夹包含未放在 `/var/lib` 中子文件夹下的变量数据。应该在该文件夹中尽量使用相对独特的文件名，以防名字空间冲突[4]。

> [3]. 此版的标准与以前版本一个很重要的不同点是要求应用程序使用/var/lib下的子文件夹。↩

> [4]. 这一层次结构应该包含当前BSD发行版中保存在 `/var/db` 下的文件。它们包括 `locate.Database` 和 `mountdtab` 及内核符号数据库。↩

/var/lock：锁文件

5.9.1. 用途

锁文件应该保存在`/var/lock`文件夹结构中。

由多个应用程序共享的设备锁或其他资源锁文件，如以前放在`/usr/spool/locks`或`/usr/spool/uucp`下的串口设备锁文件，现在必须保存在`/var/lock`中。必须使用“`LCK..`”后跟设备的基本名称的命令惯例，例如，要锁定`/dev/ttyS0`，应该创建名为“`LCK..ttyS0`”的文件[5]。

这些锁文件的内容格式必须为 HDB UUCP 锁文件格式。HDB 格式将进程标识（PID）以10字节的 ASCII 十进制数字形式保存，后面跟一个换行符。例如，如果进程1230持有一个锁文件，该文件会包含11个字符：空格、空格、空格、空格、空格、空格、空格、1、2、3、0和换行。

> 5. 然后，任何想要使用`/dev/ttyS0`的对象都可以读取这个锁文件并作出相应的反应（`/var/lock`下的所有文件都必须是全局可读的）。↩

/var/log：日志文件和文件夹

5.10.1. 用途

此文件夹中包含各种日志文件。绝大多数的日志应该写入此文件夹或其中某个相应的子文件夹。

5.10.2. 特殊选项

如果安装了相应的子系统，`/var/log` 中必须有以下文件或符号链接：

文件	描述
lastlog	每位用户上次登录的记录
messages	Syslogd产生的系统消息
wtmp	所有登录和注销的记录

/var/mail：用户邮箱文件（可选）

5.11.1. 用途

必须能够通过 `/var/mail` 访问邮件spool，并且邮件spool文件必须采用 `<usrname>` 的形式[6]。

这里的用户邮箱文件必须以标准的UNIX邮箱格式保存。

> ### 基础知识
>
> 此文件夹的逻辑位置从 `/var/spool/mail` 修改到这里是为了使FHS与几乎每个UNIX安装版相兼容。这一修改对互操作性很重要，因为人们经常在多个主机间和多个UNIX安装版间（忽略NFS锁问题）共享一个 `/var/mail` 。
>
> 要特别注意，这里没有要求将邮件spool实际移动到这一位置。然而，程序和头文件应该改成使用 `/var/mail` 这个位置。

> 6. 注意/var/mail可以是另一文件夹的符号链接。 ↩

/var/opt：/opt的变量数据

5.12.1. 用途

`/opt` 下软件包的各种变量数据应该安装在 `/var/opt/<subdir>` 中，其中 `<subdir>` 是 `/opt` 中子树的名称，那里保存了第三方软件包的静态数据，除非该子树被 `/etc` 下的另一文件所取代。对 `/var/opt/<subdir>` 下内部的组织方式没有结构性要求。

> ### 基础知识
>
> 参见/opt的基础知识部分。

/var/run：运行时可变数据

5.13.1. 用途

此文件夹包含描述系统启动以来系统信息的数据。此文件夹下的文件必须在启动过程初期清除（删除或归零）。程序可以在 `/var/run` 下有自己的子文件夹；鼓励使用多于1个运行时文件的程序使用这种方法[7]。

原先放在 `/etc` 下的进程标识（PID）文件必须放在 `/var/run` 里面。PID文件的命名惯例是 `<program-name>.pid` 。例如， `crond` 的PID文件名为 `/var/run/crond.pid` 。

5.13.2. 要求

PID文件的内部格式没有变化。文件必须包含ASCII编码的十进制进程标识，后跟一个换行符。例如，如果crond的进程号为25， `/var/run/crond.pid` 将含有三个字符：2、5、换行。

读取PID文件的程序应该对接收的内容具备一定的灵活性；例如，它们应该忽略额外的空白、开头的零字符、末尾的换行符缺失、PID文件中有多余行这些情况。创建PID文件的程序应该使用上一段中简单的规定。 Utmp文件保存了当前谁正在使用系统的信息。它位于此文件夹中。维护暂时性UNIX域套接字的系统程序必须将它们放在此文件夹下。

---

7. `/var/run` 对未授权用户（运行后台服务的root或用户）应该是不可写的；如果谁都能在这个文件夹中写入，就会成为一个重大的安全隐患。 ↩

/var/spool：应用程序spool数据

5.14.1. 用途

`/var/spool` 包含正在等待某种后续处理的数据。`/var/spool` 中的数据表示工作将（由程序、用户或管理员）在以后完成；在处理后这些数据经常会被删除[8]。

5.14.2. 特殊选项

如果安装了相应子系统，则 `/var/spool` 下必须有以下文件夹或符号链接：

文件夹	描述
lpd	打印机spool文件夹（可选）
mqueue	寄出邮件队列（可选）
news	新闻spool文件夹（可选）
rwho	Rwhod文件（可选）
uucp	UUCP的spool文件夹（可选）

5.14.3. /var/spool/lpd：线性打印机后台服务打印队列（可选）

5.14.3.1. 用途

Lpd的锁文件 `lpd.lock` 必须安装在 `/var/spool/lpd` 下。建议每个打印机的锁文件放在特定于该打印机的 `spool` 文件夹下并命名为 `lock` 。

5.14.3.2. 特殊选项

文件夹	描述
printer	某个特定打印机的spool（可选）

5.14.4. /var/spool/rwho：Rwhod文件（可选）

5.14.4.1. 用途

此文件夹保存了本地网络上其他系统的rwhod信息。

> ### 基础知识
>
> 有些BSD发行版使用`/var/rwho`存放这类数据；鉴于它们在其他系统中以前的位置是`/var/spool`并且它们基本符合‘spool’数据的定义，我们认为使用这一位置更恰当。

> [8]. UUCP锁文件必须放在`/var/lock`中。见上节关于`/var/lock`的内容。↩

/var/tmp：系统重启间保留的临时文件

5.15.1. 用途

`/var/tmp` 文件夹给那些要求在两次系统启动间保留临时文件或文件夹的程序使用。因而，`/var/tmp` 下保存的数据比 `/tmp` 中的更持久些。

系统重新启动时禁止删除 `/var/tmp` 下的文件和文件夹。尽管 `/var/tmp` 中的数据如何删除一般依站点而定，还是建议删除操作的频率间隔应少于 `/tmp` 。

/var/yp：网络信息服务（NIS）数据库（可选）

5.16.1. 用途

网络信息服务的可变数据，即原来所谓的Sun黄页（YP），必须放在此文件夹。

> 基础知识
>
> `/var/yp` 是NIS（YP）的标准文件夹，几乎只在NIS文档和系统中使用[9]。

> 9. 不要把NIS与Sun的NIS+混淆，它使用另一个文件夹，`/var/nis`。↩

操作系统特定的附加说明

本节是只针对特定操作系统的附加要求和建议。本节中的材料任何情况下不应与基本标准相冲突。

Linux

这是对Linux操作系统的附加说明。

6.1.1. `/` ：根文件夹

在Linux系统上，如果内核位于 `/` ，我们建议使用 `vmlinux` 或 `vmlinuz` 这样的名称，这在最近的内核源软件包中已经采用。

6.1.2. `/bin` ：必要的用户命令二进制文件（所有用户使用）

需要它们的Linux系统将这些附加文件放在 `/bin` 中：

- Setserial

6.1.3. `/dev` ：设备和特殊文件

`/dev` 下必须有以下文件

- `/dev/null`

  所有写入到这一设备的数据将被忽略。读该设备将返回一个EOF条件。

- `/dev/zero`

  该设备是一个输出零数据的源。所有写入此设备的数据被忽略。读该设备时将返回包含所请求数量的0值。

- `/dev/tty`

  这一设备是进程控制终端的同义词。一旦打开该设备，所有的读取和写入会和实际打开了控制终端设备一样。

> 基础知识
>
> 以前的FHS版本对 `/dev` 有更严格的要求。其他设备也可以存在于 `/dev` 中。设备名可以是到 `/dev/` 或 `/dev` 的子文件夹下其他设备节点的符号链接。没有对主/次设备号值的要求。

6.1.4. `/etc` ：主机特定的系统配置

需要它们的Linux系统将这些附加文件放在 `/etc` 。

- lilo.conf

6.1.5. /lib64和/lib32：64位/32位库（依赖于架构）

64位架构的处理器PPC64、s390x、sparc64和AMD64必须将64位库放在 `/lib64` 中，32位（或s390上的31位）库放在 `/lib` 中。

64位架构IA64必须将64位库放在 `/lib` 中。

> 基础知识
>
> 这是对 `/lib<qual>` 和 `/usr/lib<qual>` 通用规则的一个完善。PPC64、s390x、sparc64和AMD64架构同时支持32位（对于s390更准确是31位）和64位程序。使用 `/lib` 放置32位二进制文件可以使现有32位系统中的二进制不须作任何改变就可以使用：这些二进制文件是很多的。IA-64使用不同的方案，反映了该架构对32位二进制文件（连带库）的排斥。

6.1.6. `/proc` ：内核和进程信息虚拟文件系统

Proc是Linux中处理进程和系统信息的实际标准方法，而不是 `/dev/kmem` 和其他类似的方法。我们强烈建议使用它来存储和取回进程信息和其他内核与内存信息。

6.1.7. `/sbin` ：必要的系统二进制文件

Linux系统将这些附加文件放在 `/sbin` 中。

- 第二扩展文件系统（ext2）命令（可选）：
  - `badblocks`
  - `dumpe2fs`
  - `e2fsck`
  - `mke2fs`
  - `mklost+found`
  - `tune2fs`
- 引导装载程序映射安装器（可选）：
  - `lilo`

其他/sbin的文件：

- 静态二进制文件：
  - `ldconfig`
  - `sln`
  - `ssync`

使用静态的 `ln` （ `sln` ）和静态的 `sync` （ `ssync` ）在出错时很有用。 `Sln` 的主要用途（在进行编排不当的升级后修复 `/lib` 中受损的符号链接）已经不是关注的重点；因为有了 `ldconfig` 程序（通常位于 `/usr/sbin` ），它可以作为升级动态库的向导工具。在一些紧急情况下静态的 `sync` 是有用的。注意这不是说必需使用静态链接版本的 `ln` 和 `sync` ，但可以是。

`ldconfig` 二进制文件是 `/sbin` 中可选的，因为一个站点可能选择在启动时运行 `ldconfig` ，而不只在升级共享库时。（在每次启动时运行 `ldconfig` 是好是坏没有定论）。尽管如此，有些人喜欢顺手使用 `ldconfig` ，在以下情况下（都很常见）：

1. 刚删除了 `/lib/<file>` 。
2. 找不到库的名称，因为 `ls` 是动态链接的，使用了一个没有内建 `ls` 的shell，也不知道可以用 `echo *` 代替它。
3. 有一个静态的 `sln` ，但不知道如何调用该链接。

- 杂项：
  - `Ctrlaltdel`
  - `Kbdrate`

  这里为了应付有些键盘自带的重复速度过快以致无法使用的情况，可以在一些系统中安装 `kbdrate` 程序到 `/sbin` 。

  由于内核中 `Ctrl` - `Alt` - `Del` 组合键的默认动作是立即硬重启，通常可以考虑在以读写模式挂载根文件系统前禁止该行为。有些 `init` 套件能够禁用 `Ctrl` - `Alt` - `Del` ，但另一些可能需要 `ctrlaltdel` 程序，它可以安装在这些系统的 `/sbin` 中。

6.1.8. /usr/include：C 程序包含的头文件

对于安装了C或C++编译器且不使用glibc的系统，要求有以下的符号链接。

- `/usr/include/asm` -> `/usr/src/linux/include/asm-<arch>`
- `/usr/include/linux` -> `/usr/src/linux/include/linux`

6.1.9. /usr/src：源代码

对于基于glibc的系统，对此文件夹没有特殊规定。对于基于比glibc早的Linux libc修订版的系统，要遵守以下的规定和基本原理：

- 唯一应该被放置在特殊位置的是Linux内核源代码。它位于/usr/src/linux。
- 如果安装了C或C++编译器，但没有安装完整的Linux内核源代码，则内核源代码的头文件必须位于以下文件夹：
  - `/usr/src/linux/include/asm-<arch>`
  - `/usr/src/linux/include/linux`

`<arch>`是系统架构的名称。

> 注意
>
> `/usr/src/linux`可以是到内核源码树的符号链接。
>
> 基本知识
>
> 内核头文件放在`/usr/src/linux`而不是`/usr/include`是很重要的，因为这样系统管理员初次更新他们的版本号时不会出现问题。

6.1.10. `/var/spool/cron`：`cron`和`at`任务

此文件夹包含了`cron`和`at`程序的可变数据。

附录

FHS邮件列表

FHS邮件列表位于freestandards-fhs-discuss@lists.sourceforge.net。您可以在这个页面http://sourceforge.net/projects/freestandards订阅它。

感谢位于SanDiego哥伦比亚大学的网络运营部门，他们允许我们使用其优良的邮件列表服务器。 正如在介绍中提到的，在与FHS编辑或列出的贡献者沟通之前，请不要往邮件列表里发邮件。

FHS的背景

开发一个标准文件系统层次结构的工作始于1993年8月，旨在重建Linux的文件和文件夹结构。FSSTND，即Linux操作系统特定文件系统层次结构标准，于1994年2月14日发布。后来又在1994年10月9日和1995年3月28日发布了两个修订版。

1995年年初，在BSD开发社区成员的帮助下，发展一个更详尽的、不仅面向Linux也面向其他类UNIX系统的FSSTND版本的目标被提出来。结果，人们针对类UNIX系统中常见的问题集中作了一些协调性的努力。鉴于目标的扩大，该标准的名字修改为文件系统层次结构标准，简称FHS。

为标准作出大量努力的志愿者列出文档末。本标准是这些人和其他贡献者合力展现的成果。

总体原则

这里是一些在发展此标准时所使用的总体原则：

- Solve technical problems while limiting transitional difficulties。
- Make the specification reasonably stable。
- Gain the approval of distributors,developers,andotherdecision-makers in relevant development groups and encourage their participation。
- Provide a standard that is attractive to the implementors of different UNIX-like systems。

适用范围

本文档通过规定文件和文件夹的位置和一些系统文件的内容，为FHS制了一个标准的文件系统层次结构。 本标准的设计目标用户是构造和维护FHS兼容文件系统的系统集成者、包开发者和系统管理员。其初衷是作为一本参考，而不是如何管理一个合格的文件系统层次结构的教材。

FHS是从早期FSSTND工作的基础上发展起来的。它在FSSTND基础上构建，关注了Linux社区及更广范围内，包括基于4.4BSD的操作系统上的互操作性问题。它吸收了BSD体系和其他地方有关多架构支持和多元化网络方面的经验。

尽管本标准与以前对文件系统层次结构标准化的努力相比更加详尽，由于新技术的出现，相关的改进导致的周期性更新也是必要的。也有可能这里讨论的问题已经有了更好的解决方案，以致我们的解决方法已经不再是最好的可能方案。我们可能在周期性更新之外发布补充草案。然而，文档一个版本到下一版本的向后兼容性是一个明确的目标。

欢迎对标准提意见。任何改进的意见和建议都可以向FHS编辑（DanielQuinlan<quinlan@pathname.com）或FHS邮件列表提出。排版和语法建议应该向FHS编辑提出。

在向邮件列表发送邮件之前，建议您首先联系FHS编辑以多余的重复讨论旧话题。

偶尔会有如何理解文档中某些条目的问题出现。如果您需要释疑，请联系FHS编辑。由于本标准代表了很多参与者的协作成果，确保任何的解释也代表了他们的集体意见是很重要的。因此，立即得到回复可能不太现实，除非这条质询是以前讨论过的话题。

致谢

FHS的开发者想感谢那些对本标准作出必不可少的贡献的开发者、系统管理员和用户。我们想感谢每位帮助编写、编译、和排版的贡献者。

FHS也要感谢那些支持本标准的旧版FSSTND的Linux开发者，如果不是他们证明了FFSTND的益处，FHS也不会得到发展。

贡献者

姓名	联系方式
BrandonS.Allbery	bsa@kf8nh.wariat.org
KeithBostic	bostic@cs.berkeley.edu
DrewEckhardt	drew@colorado.edu
RikFaith	faith@cs.unc.edu
StephenHarris	sweh@spuddy.mew.co.uk
IanJackson	ijackson@cus.cam.ac.uk
AndreasJaeger	aj@suse.de
JohnA.Martin	jmartin@acm.org
IanMcCloghrie	ian@ucsd.edu
ChrisMetcalf	metcalf@lcs.mit.edu
IanMurdock	imurdock@debian.org
DavidC.Niemi	niemidc@clark.net
DanielQuinlan	quinlan@pathname.com
EricS.Raymond	esr@thyrsus.com
RustyRussell	rusty@rustcorp.com.au
MikeSangrey	mike@sojurn.lns.pa.us
DavidH.Silber	dhs@glowworm.firefly.com
ThomasSippel-Dau	t.sippel-dau@ic.ac.uk
TheodoreTs’o	tytso@athena.mit.edu
StephenTweedie	sct@dcs.ed.ac.uk
FredN.vanKempen	waltje@infomagic.com
BerndWarken	bwarken@mayn.de
ChristopherYeoh	cyeoh@samba.org

> 翻译者： Wylmer Wang